Remarks

The claims have been amended to refer to "one or more" transfer points rather than to "a plurality of" transfer points as before, since patentability of the base claims is not predicated on the particular number of transfer points.

Claims

Claims 1-4 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over U.S. Patent 5,202,995 to O'Brien ("O'Brien") in view of U.S. Patent 6,513,156 to Bak et al. ("Bak") and Bacon et al., "Compiler Transformations for High-Performance Computing", ACM Computing Surveys, vol. 26, no. 4, Dec. 1994 ("Baker") (paper no. 9, page 3). This rejection is respectfully traversed.

O'Brien describes a method for removing invariant branches from instruction loops of a computer program. This technique is an optimization to move a loop invariant branch in a reducible loop, called SESCE in the disclosure, to outside of the loop. As the patentee notes at column 3, lines 33-41 and 49-53:

> In the process of optimizing a computer program, it may be desirable to relocate a conditional branch instruction to a point outside of its originally containing loop. This may occur, for example, when a conditional branch instruction demonstrates the property of loop invariance, which is to say, it is evaluated identically regardless of the iteration of the loop. The relocation of such a statement to a point just prior to loop execution alleviates repetitious, unnecessary evaluations. . . .
>
> In the exemplary embodiment of the invention described below, a loop that has an invariant conditional branch instruction is transformed into two loops. The first loop is the original loop. The second loop is an exact copy of the original loop.

As shown in Figs. 9A-9B and accompanying text, he accomplishes this by copying a loop (nodes B-F) containing an invariant conditional branch (node B), moving the conditional branch instruction to a location (node A') before either loop, and having that moved instruction branch to one or the other of the two loops, depending on the value of the condition.

While O'Brien is thus generally relevant to loop optimization, his object is a particular type of loop optimization, not preparing a loop for general forms of optimization by making the loop irreducible. Thus, O'Brien does not teach moving transfer points to the top of a loop process, as claimed by applicants. Indeed, in the example shown in Fig. 9A, there are no transfer points at all inside the loop process consisting of nodes B-F, only the original entry point (node B) at the top of the loop.[1] Since there are no such internal transfer points, they clearly cannot be moved to the top of a loop process as claimed by applicants.

Not only does O'Brien not teach relocating a transfer point located in a loop other than at the top of the loop, but it is incapable of being used in such a manner as well. This is because such a transfer point changes the original reducible loop into an irreducible loop, and the target loop of the technique is restricted to a reducible loop, as described at column 4, lines 10-17. Nor can O'Brien's method transform an irreducible loop into a reducible loop, as it merely removes a loop invariant branch or replaces it with an unconditional branch; the entry points of the loop are not removed or moved to outside of the loop by this process.

O'Brien also does not teach copying code from the top of a loop process to a point that post-dominates the top of the loop process and a transfer point to a location immediately preceding the loop process if the transfer point is located inside the loop process, as further claimed by applicants. Since, as noted above, there is no transfer point at all inside the loop process consisting of nodes B-F, the condition for performing this claimed step of applicants' simply doesn't occur in O'Brien.

Given this basic failure of the primary reference of O'Brien to teach what it is asserted to teach—namely, (1) moving a transfer point to the top of a loop process if it can be moved there without a problem occurring and (2) copying code from the top of the loop process to a point that post-dominates the top of the loop process and the transfer point to a location immediately preceding the loop process if the transfer point is located inside the loop process—it hardly matters what the secondary references teach. In any event, Bak merely teaches switching from

[1] If the conditional branch instruction B in Fig. 9A is deemed a transfer point, it is so only because it is at the top of the loop.

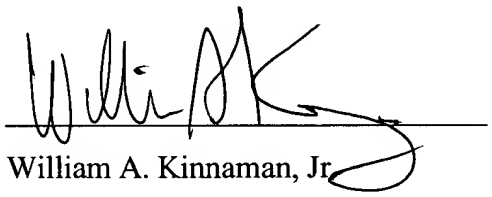
virtual machine mode to native execution mode for heavily traversed code segments, something that applicants are not claiming per se. Similarly, Baker merely teaches certain well-known loop optimization techniques that applicants are not claiming per se.

For the foregoing reasons, claims 1-4, both as previously presented and as amended, clearly distinguish patentably over the art cited by the Examiner.

Conclusion

Reconsideration of the application as amended is respectfully requested. It is hoped that upon such consideration, the Examiner will hold all claims allowable and pass the case to issue at an early date. Such action is earnestly solicited.

Respectfully submitted,
TOSHIAKI YASUE et al.

By 

William A. Kinnaman, Jr.
Registration No. 27,650
Phone: (845) 433-1175
Fax: (845) 432-9601

WAK/wak